In the Specification

Please amend the specification of this application as follows:

Rewrite the paragraph at page 1, lines 4 to 5 as follows:

B2 --Serial Number 09/154,385 entitled "METHOD OF INITIALIZING A CPU CORE FOR EMULATION" filed September 16, 1998, now U.S. Patent No. 6,167,385 issued December 26, 2000; and--

Rewrite the paragraph at page 1, lines 7 to 9 as follows:

B3 --Serial Number ~~(TI-28928P)~~ 09/483,367, entitled "EMULATION SUSPEND MODE WITH DIFFERING RESPONSE TO DIFFERING CLASSES OF INTERRUPTS" claiming priority from U.S. Provisional Application No. 60/120,809 filed February 19, 1999, now U.S. Patent No. 6,553,513;--

Rewrite the paragraph at page 1, lines 10 to 11 as follows:

B4 --Serial Number ~~(TI-28929P)~~ 09/481,852, entitled "EMULATION SUSPENSION MODE WITH STOP MODE EXTENSION" claiming priority from U.S. Provisional Application No. 60/120,809 filed February 19, 1999, now U.S. Patent No. 6,567,933;--

Rewrite the paragraph at page 1, lines 12 to 13 as follows:

B5 --Serial Number ~~(TI-28930P)~~ 09/483,568, entitled "EMULATION SUSPEND MODE HANDLING MULTIPLE STOPS AND STARTS" claiming priority from U.S. Provisional Application No. 60/120,809 filed February 19, 1999, now U.S. Patent No. 6,564,339;--

Rewrite the paragraph at page 1, lines 14 to 15 as follows:

B6 --Serial Number ~~(TI-28931P)~~ 09/483,697, entitled "EMULATION SUSPEND MODE WITH FRAME CONTROLLED RESOURCE ACCESS " claiming priority from U.S. Provisional Application No. 60/120,809 filed February 19, 1999, now U.S. Patent No. 6,557,116;--

Rewrite the paragraph at page ~~1~~, lines 16 to 17 as follows:

--Serial Number ~~(TI-28932P)~~ 09/482,902, entitled "EMULATION SUSPEND MODE WITH INSTRUCTION JAMMING" claiming priority from U.S. Provisional Application No. 60/120,809 filed February 19, 1999;--

Rewrite the paragraph at page ~~1~~, lines 18 to 19 as follows:

--Serial Number ~~(TI-28933P)~~ 09/483,570, entitled "SOFTWARE EMULATION MONITOR EMPLOYED WITH HARDWARE SUSPEND MODE" claiming priority from U.S. Provisional Application No. 60/120,683 filed February 19, 1999;--

Rewrite the paragraph at page ~~1~~, lines 20 to 22 as follows:

--Serial Number ~~(TI-28934P)~~ 09/438,237, entitled "EMULATION SYSTEM ~~WITH~~ SEARCH AND IDENTIFICATION OF OPTIONAL EMULATION PERIPHERALS" claiming priority from U.S. ~~Provisional~~ Application No. 60/120,960 filed February 19, 1999;--

Rewrite the paragraph at page 1, lines 23 to 25 as follows:

--Serial Number ~~(TI-28935P)~~ 09/483,783, entitled "EMULATION SYSTEM WITH ADDRESS COMPARISON UNIT AND DATA COMPARISON UNIT OWNERSHIP ARBITRATION" claiming priority from U.S. Provisional Application No. 60/120,791 filed February 19, 1999, now U.S. Patent No. 6,606,590; and--

Rewrite the paragraph at page 1, lines 26 to 28 as follows:

--Serial Number ~~(TI-28937P)~~ 09/483,321 entitled "EMULATION SYSTEM EMPLOYING SERIAL TEST PORT AND ALTERNATIVE DATA TRANSFER PROTOCOL~~.~~" claiming priority from U.S. Provisional Application No. 60/120,667 filed February 19. 1999.--

Rewrite the paragraph at page 3, line 24 to page 4, line 6 as follows:

--Another problem is product emulation when employing these programmable digital processors. Product development and debugging is best handled with an emulation circuit closely corresponding to the actual integrated circuit to be employed in the final product. In circuit emulation (ICE) is in response to this need. An integrated circuit with ICE includes auxiliary ~~circuit~~ circuits not needed in the operating product included solely to enhance emulation visibility. In the typical system level integration circuit, these emulation circuits use only a very small fraction of the number of transistors employed in operating circuits. Thus it is feasible to include ICE circuits in all integrated circuits manufactured. Since every integrated circuit can be used for emulation, inventory and manufacturing need not differ between a normal product and an emulation enhanced product.--

Rewrite the paragraph at page 5, lines 2 to 18 as follows:

--This invention is in-circuit-emulation of an integrated circuit including a digital data processor capable of executing program instructions. A first debug event is detected during normal program execution. The causes the in-circuit-emulation to suspend program execution except for real time interrupts. A debug frame counter increments on each interrupt and decements on each return from interrupt. If a debug event is detected during an interrupt service routine, that interrupt service routine is suspended and ~~teh~~ the count of the debug frame counter is stored. Execution of other interrupt service routines in response to corresponding interrupts is still permitted. The integrated circuit includes plural debug event detectors and the debug frame count is stored at the detector detecting a debug event during an ~~interrput~~ interrupt service routine. This permits a determination

of the order of interrupts triggering debug events by reading the stored debug frame count from each debug event detector.--

Rewrite the paragraph at page 10, lines 6 to 17 as follows:

--The preferred embodiment of this invention includes an extension to the JTAG interface. Two pins nET0 and nET1 serve as a two pin trigger channel function. This function supplements the serial access capability of the standard interface with continuous monitoring of device activity. The two added pins create debug and test capabilities that cannot be created with the standard interface. The nET0 signal is called Emulation and Test 0 Not. This signal helps create a trigger to channel zero. Similarly, the nET1 signal is called Emulation and Test ~~0~~ 1 Not. This signal helps create a trigger to channel one. These channels will be further explained below.--

Rewrite the paragraph at page 10, lines 18 to 26 as follows:

--Figure 3 illustrates an emulation level view of target system 3. Target system 3 may include plural devices 11, 13 and 15. Figure 3 illustrates details of example device 13 which includes plural megamodules 21, 23 and 25. Figure 3 illustrates details of example ~~megamodules~~ megamodule 23. Example megamodule 23 includes debug and test control unit 30 and plural device domains. These device domains include central processing unit (CPU) core 31, analysis unit 33, memory 35 and debug/test direct memory access (DT_DMA) unit 37.--

Rewrite the paragraph at page 11, lines 3 to 15 as follows:

--Figure 4 illustrates an electrical connection view of the coupling ~~between access~~ adapter 2 and target system 3. Figure 4 shows the connections of the ~~of the various signals~~ of the JTAG header 5 illustrated in Figure 2. All these signals are connected

to scan controller 41. The signals nTRST, TCK and TMS are connected to two example megamodules 31 and 33. Figure 4 illustrates the optional connection of TCKO to the target system clock SYSCLK. The scan input TDI connects to a scan input of megamodule 31. The scan output of megamodule 31 supplies the scan input of ~~eg module~~ megamodule 33. The scan output of meg module 33 supplies the scan output TDO. The two extension signals nET0 and nET1 control ~~meg modules~~ megamodules 31 and 33 via merge unit 32. These extension signals are monitored by test equipment 43.--

Rewrite the paragraph at page 11, lines 17 to 24 as follows:

--The debugging environment illustrated in Figures 1 to 4 permit the user to control application execution by any programmable digital processor of target system 3. Typical control processes include: keyboard directives such as run, halt and step; software ~~breakpoint~~ breakpoints using op-code replacement; internal analysis ~~breakpoint~~ breakpoints specified program counter or watchpoints specified by data accesses; and externally generated debug events.--

Rewrite the paragraph at page 11, line 24 to page 12, line 2 as follows:

--Actions such as decoding a software breakpoint instruction (DSTOP), the occurrence of an analysis breakpoint or watchpoint (ASTOP), or the occurrence of a debug host computer event (HSTOP) are referred to as debug events. Debug events cause execution to suspend. Debug events tied to the execution of specific instructions are called ~~breakpoint~~ breakpoints. Debug events generated by memory references are called watchpoints. External debug events can also suspend execution. Debug events cause entry into the Debug State.--

Rewrite the paragraph at page ~~12~~, lines 8 to 16 as follows:

--Execute state 101 corresponds to the ordinary operation of target device 3. In the execute state 101 instructions are executed by the programmable digital processor in normal fashion. There are no outstanding debug suspend conditions. A low logic level applied to the nTRST terminal or a software directive requesting functional run forces the operational state to execute state 101. In execute state 101 the pipeline fetches and executes instructions and ~~process~~ processes interrupts in a normal way.--

B19

Rewrite the paragraph at page 13, lines 4 to 14 as follows:

--In the debug suspend state 102 background instruction execution is inactive. This state permits debug/emulation visibility into the state of target device 3 while background execution is suspended. In debug suspend state 102, the program counter (PC) and status bits are generally preserved at their values prior to the debug event. The PC points to the instruction to be executed next. When execution resumes, the instruction referenced by the PC and those following ~~is~~ are fetched for execution. This is facilitated by flushing the front end of the pipeline upon entry into debug suspend state 102 from execute state 101.--

B20

Rewrite the paragraph at page 15, lines 1 to 9 as follows:

--The digital frame counter is decremented upon each return from interrupt. This count permits the debug environment to track the status of the suspended foreground task. For example, a taken high priority interrupt may change the machine state and thus the current machine state would not reflect the suspended background task. However, if the digital frame counter were zero, then the debug environment is assured ~~no~~ that interrupts have not temporarily changed the machine state.--

Sale

B21

Rewrite the paragraph at page 17, lines 5 to 18 as follows:

--Figure 7 illustrates in greater detail circuits located on each megamodule concerned with emulation. These include address comparison unit (ACU) 310, data comparison unit (DCU) 320 and the external ~~control~~ comparison unit 330 (ECU). The address comparison unit 310 provides breakpoint, counter, parallel signature analysis and data logging support. The data comparison unit 320 provides breakpoint and parallel support. The external comparison unit 330 controls external inputs to the event functions. Interaction with the programmable digital processor within the megamodule is handled by the memory unit 301. The application and debug software share access to address comparison unit 310, data comparison unit 320 and external comparison unit 330 by access to their registers.--

Rewrite the paragraph at page 17, line 25 to page 18, line 8 as follows:

--Address comparison unit 310 contains two 32 bit registers AREF and AMSK and one 16 bit register ACNTL. The AREF and AMSK registers are preferably 32 bit data registers that can be addressed as sixteen bit registers in 16 bit architectures. Their function is defined by the ACNTL register described in Table 2. The ACNTL register configures the AREF and AMSK registers in a number of modes, including: DMA reads and writes for data logging, downloads and uploads; event generation such as breakpoints, watchpoints and nET0 and nET1 triggers; counts for benchmarking, watchdog timing and period counters; parallel signature analysis functions for test; off ~~preforming~~ performing no function and ownership by the application or debug is unchanged; and unclaimed performing no function and either the application or debug can obtain ownership. Address comparison unit 310 is responsive to the bus input selected by multiplexer 311.--

Rewrite the paragraph at page 18, lines 9 to 22 as follows:

--The address comparison unit 310 configures for event generation where the AMSK register serves as an address mask register and the a AREF register serves as an address reference. The address comparison unit 310 generates a debug suspend request when the ACNTL register ASTOP ~~and AFEN bits are~~ is TRUE. The AMSK field defines the address comparison unit 310 debug suspend request rudeness level. The ability to generate ~~and~~ an event without generating a debug suspend request allows the address comparison unit 310 event to be used as a trigger generator through the ETx pins without altering core execution. This function supports breakpoints, watchpoints, and trigger generation. Table 2 shows the function specific bit mode bit definition of register ACNTL for event generation.--

Rewrite the paragraph at page 22, line 23 to page 23, line 9 as follows:

--The data comparison unit ~~310~~ 320 contains two 32 bit registers DREF and DMSK and one 16 bit register DCNTL. The DREF and DMSK registers are merely 32 bit data registers that can be addressed as sixteen bit registers in 16 bit architectures. Their function is defined by the DCNTL register described in Table 5. This DCNTL register configures the DREF and DMSK registers in a number of modes, including: event generation such as breakpoints, watchpoints and nET0 and nET1 triggers; parallel signature analysis functions for test; reloadable period counts; off ~~preforming~~ performing no function and ownership by the application or debug is unchanged; and unclaimed performing no function and either the application or debug can obtain ownership. Data comparison unit 320 is responsive to the bus input selected by multiplexer 321.--

Rewrite the paragraph at page 23, lines 10 to 21 as follows:

--The data comparison unit ~~310~~ 320 configures for event generation where the DMSK register serves as a mask register and the ~~a~~ DREF register serves as a comparison reference. The data comparison unit ~~310~~ 320 generates a debug suspend request when the DCNTL register DSTOP and DFEN bits are TRUE. The DMSK field defines the data comparison unit ~~310~~ 320 debug suspend request rudeness level. Generation of an event without generating a debug suspend request allows the data comparison unit ~~310~~ 320 event to be used as a trigger generator through the nET0 and nET1 pins without altering core execution. This function supports break and watchpoints, execution pause, and event counting.--

Rewrite the paragraph at page 23, lines 22 to 30 as follows:

--The data comparison unit ~~310~~ 320 event generation works in tandem with the address comparison unit 310 event generation to provide address and data breakpoints. This feature requires that the two units be joined. The address comparison unit 310 event detects the address match while the data comparison unit ~~310~~ 320 detects the read data or write data match associated with an access. The address comparison unit 310 address comparison is delayed to align with the data comparison unit ~~310~~ 320 event processing.--

Rewrite the paragraph at page 23, line 31 to page 24, line 6 as follows:

--The data comparison unit ~~310~~ 320 provides a unique ability to compare up to 32 user supplied inputs to a reference. The user inputs supplied to the megamodule ~~Core~~ core can be parallel signature analyzed or used as events. The selection of the data comparison unit ~~310~~ 320 parallel signature analysis mode is made available to the logic outside the CPU megamodule. Table 5 shows

the function specific bit mode bit definition for data comparison unit ~~310~~ 320 event generation.--

Rewrite the paragraph at page 26, lines 1 to 5 as follows:

--The data comparison unit ~~310~~ 320 counter function, when implemented, are identical to those for the address comparison unit 310. Please refer to Table 3 for a description of counter modes. The input from the address comparison unit 310 is named DAEVT instead of ADEVT.--

Rewrite the paragraph at page 26, lines 6 to 17 as follows:

--The data comparison unit ~~310~~ 320 configures for a parallel signature analysis functions where the DMSK and DREF registers serve a parallel signature analysis generator. The data comparison unit ~~310~~ 320 parallel signature analysis function provides for the selection of any of the six sources shown in Table 6 as the parallel signature analysis input. The parallel signature analysis calculation begins when the parallel signature analysis function is enabled and terminates when the data comparison unit ~~310~~ 320 function is specified as OFF. Changing the function to another function has undetermined results. Table 6 shows the function specific bit mode bit definition for parallel signature analysis functions.--

Rewrite the paragraph at page 27, lines 1 to 3 as follows:

--The data comparison unit ~~310~~ 320 configures in off and unclaimed modes identical to that of the address comparison unit 310.--

Rewrite the paragraph at page 27, lines 4 to 9 as follows:

--The external control register 330 includes a register ECNTL that manages external events that can generate debug suspend

requests. The ECNTL register manages emulation and test pin zero and one inputs as well as external input used by the logic for external hardware triggering. Refer to Table 7 for a description of this function.--